LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY

-LIGO-

CALIFORNIA INSTITUTE OF TECHNOLOGY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Document Type	DCC Number	
Test procedure	LIGO- T070263-A-C	October 22, 2007

Prototype ADC Board Design Notes & Performance Results

Joshua R. Myers

California Institute of Technology
LIGO Project – MS 18-33
Pasadena, CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project – MS 20B-145
Cambridge, MA 01239
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

http://www.ligo.caltech.edu/

1 Overview

The following document refers to D060535, the ADC Test Board. This is a prototype analog to digital converter board being developed for possible use in Advanced LIGO. A previous version (D060118) incorporating an ADC block, a DAC block, and an FPGA was built by D. Sigg, P. Schwinberg, and J. Myers and tested during the summer of 2006 by SURF student Rachel Reddick. The results of her project are available in the DCC under T060203.

This version scales back the configurability of the previous design and focuses on optimizing the noise and performance of the board. This board is meant to work in conjunction with D060293, the DAC1794A Board, being developed by P. Schwinberg and T060212, the converter uplink board, being developed by D. Sigg. It follows the specifications laid out in T060082, the converter backplane, for compatibility with the other ongoing efforts.

The general design goals for this board[1] [2] are:

- 16 channels
- ±10V differential input range (40Vpp)[3]
- 100-300nV/rtHz input referred noise
- THD < -90dB
- Crosstalk < -120dB
- CMRR > 80dB

2 ADC Selection

There is a wide variety of analog to digital converter chips available but the two primary types with enough resolution for LIGO's purposes (16 bits or greater) are sigma-delta converters and successive approximation converters. Successive approximation converters generally have lower power consumption and lower pipeline delay than sigma-delta converters and are more appropriate for use in real time digital control systems. Other desirable features are a serial interface for easier PCB layout, bipolar inputs to eliminate the need to level shift the signal and reduce the component count, a wide input range to ease the design constraints on the front end, and a high signal to noise ratio. The AD7634 from Analog Devices has all of these features and was chosen for that reason.

3 Front End Design

A 200 kHz, 5th order, type 1 Chebyshev filter with 0.5dB pass band ripple will be used as the antialiasing filter. This topology meets all of the general requirements laid out in T010066, Proposed LIGO VME ADC Specifications, except for the attenuation at the Nyquist frequency. With additional digital filtering it should be possible to meet this

[1] T010066, Proposed LIGO VME ADC Specifications
[2] T050042, Proposal for a New Converter Design
[3] Appendix 1

requirement as well. This filter has already been used successfully on D060118, the Converter Test Board, and its response is shown below.

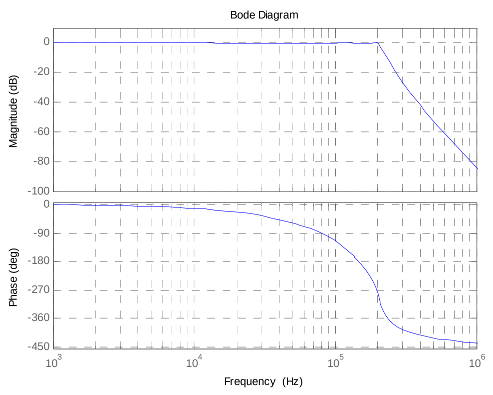


Figure 1

The full architecture is shown below. Each section will be discussed in further detail but this gives a general overview of what the overall layout will be. The input range of the AD7634 is 40Vpp differential so the input filter has unity gain in its pass band. Stage three and four drive the positive and negative inputs of the ADC and result in a gain of two so stage two will have a gain of ½ to compensate that.

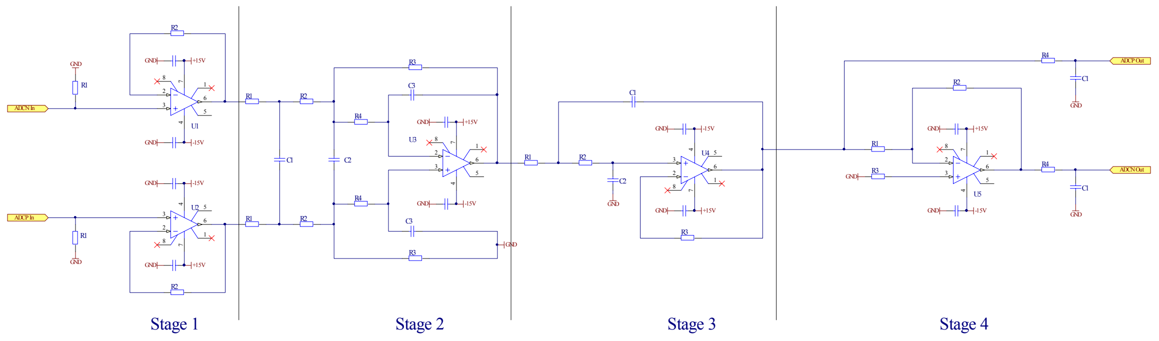


Figure 2

3.1 Stage 1, 40V p-p Differential Input

This stage (Figure 3) is the input receiver stage and buffers the input signal for the rest of the filter. Resistors R1 reference the inputs to board ground and provide a path for bias currents when there is no signal present and resistors R2 in the feedback path limit problems that might arise due to slew rate limitations. This is one of the few places where real power savings can be achieved through the use of low power op amps with minimal impact on the performance. Some of the amps that will be tested here are the AD829, the OPA134 and the LT1351. The results of those tests will be reported in another note.

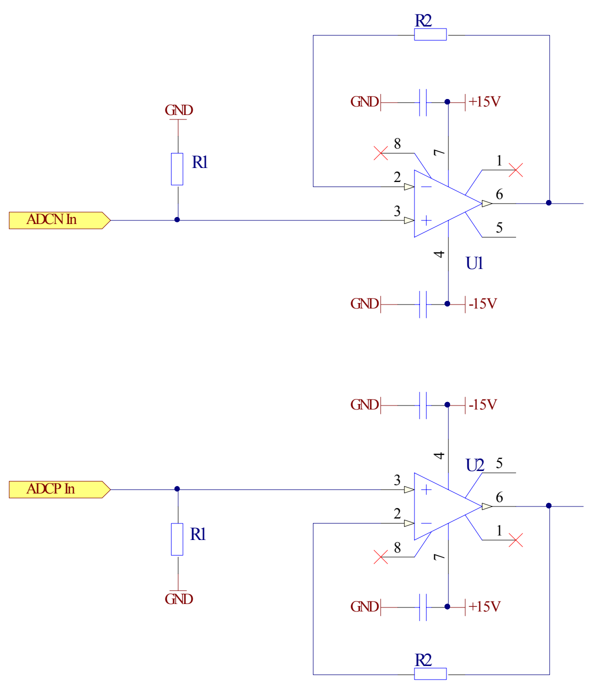
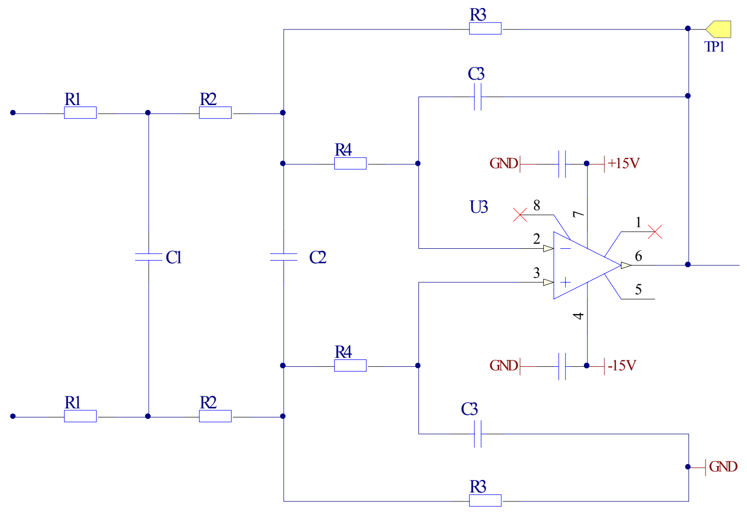


Figure 3

3.2 Stage 2, Differential 3rd Order Filter Section

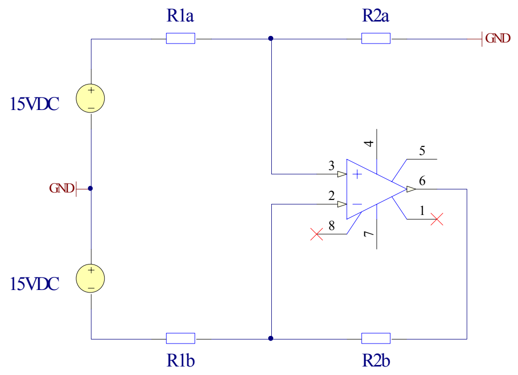


$$H(s) = \frac{1}{4\,C1\,C2\,C3\,R4\,R^2 s^3 + (2\,C1\,C3\,R^2 + 4\,C1\,C3\,R4 + 4\,C2\,C3\,R4\,R)\,s^2 + (2\,C3\,R + 3\,C3\,R4 + 2\,C1\,R)\,s + 2}$$

$R1 = R2 = R3 = R$

Figure 4

This stage (Figure 4) converts the differential signal from the input stage to a single ended signal and provides the first filtering. The differential multi-feedback topology was chosen for its superior high frequency performance and because a differential receiver was needed. The same op amps used in the first stage will also be tested here.



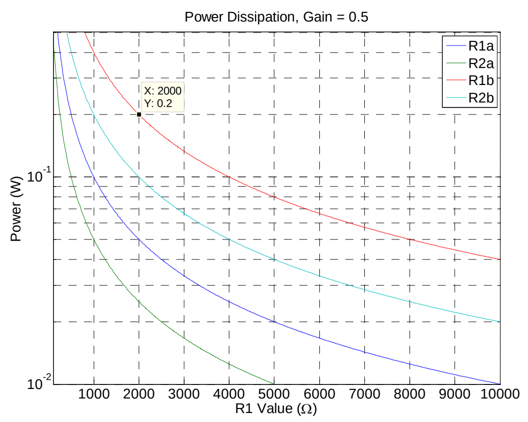
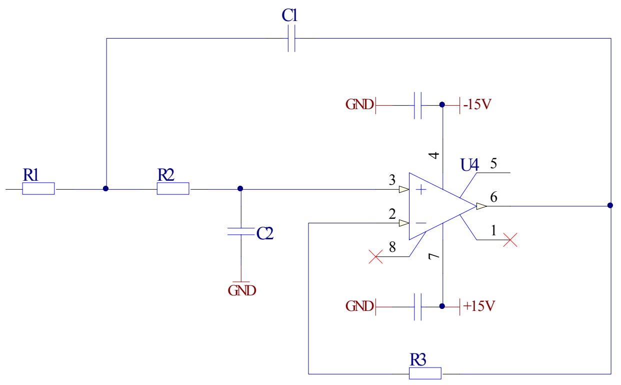


Figure 5

Johnson noise in the resistors will be minimized by choosing a low value for them but the power dissipation sets a lower limit on the size that can be used. The graph shows the expected power dissipation assuming a gain of 0.5 and a differential input structure. The resistors that will be used in this circuit are rated for 1/10W so assuming that two resistors in series are used for R1a and R1b the smallest resistors that can safely be used are 1kΩ.

3.3 Stage 3, 2nd Order Filter Section



$$H(s) = \frac{1}{C1\,C2\,R1\,R2\,s^2 + C2\,(R1 + R2)\,s + 1}$$

Figure 6

This stage (Figure 6) will be driving the ADC and will use the AD829 recommended on the ADC's data sheet. The other recommended amps don't offer any significant advantages over the AD829 and the AD829 is already used extensively in LIGO. The Sallen-Key topology does not offer the high frequency performance that the voltage feedback topologies do but the reduction in complexity and power consumption make this a suitable choice for this stage. The low pass filtering in the prior stage also reduces the high frequency requirements on this stage.

3.4 Stage 4, Inverting Buffer & Differential Output

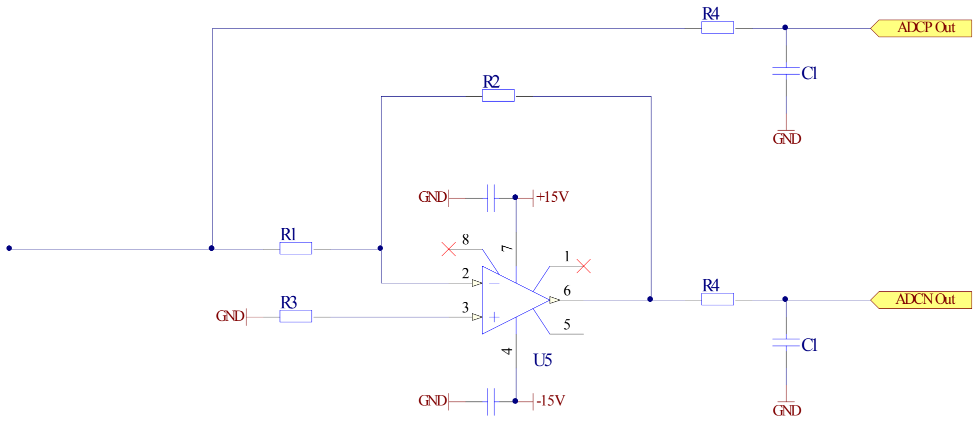


Figure 7

The final stage provides the -1 gain to develop the differential drive to the ADC and will also use the AD829. Short of a catastrophic failure in the previous stage the largest expected voltage at the input to this stage is 13.3VDC[4]. If 2k resistors are used here the maximum expected power dissipation is 88mW, well below the 100mW limit. The RC filter formed by R4 and C1 is recommended in the ADC's data sheet for preserving the SNR and transition noise performance.

4 Test Results

4.1 General Setup

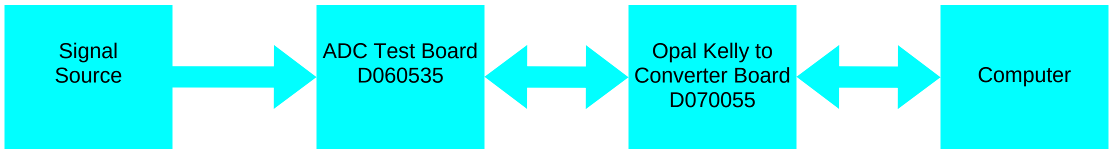


Figure 8

Each of the following tests used the same general setup shown above. The C++ code and FPGA code are lightly modified versions of the code Rachel Reddick used during her SURF project. See T060203 for more details. The filter realized in the FPGA is a 6 pole elliptic low pass filter with a cutoff frequency of 8kHz. The Matlab code to generate each of the plots is included at the end of this report.

[4] AD829 Data Sheet

4.2 Noise Performance

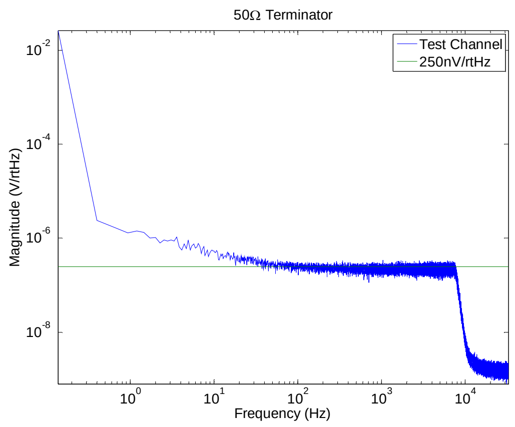


Figure 9

The theoretical noise floor for the AD7634 with a 40Vpp input range, a signal to noise ratio of 101 dB, and a sampling frequency of 524kHz is *246nV/rtHz*. This agrees well with the measured noise floor shown in the above plot. The mean noise level between 1kHz and 6kHz in the above plot is 223.5nV/rtHz, corresponding to a SNR of 101.78dB. Some investigations into why the noise below 40Hz begins to rise were made but no definitive cause has been found. In one test the inputs of the ADC chip were shorted together and this noise was still present so the input chain is not causing this behavior.

$$V_{nt} = \frac{V_{FSrms}}{10^{SNR/20}\sqrt{f_{nyq}}} = \frac{40/2\sqrt{2}}{10^{101/20}\sqrt{262144}} = 246nV/rtHz$$

In the next set of plots a series of 100Hz sine wave with amplitudes varying between 1Vpp to 39Vpp were injected into the first channel and the noise floor was measured. As the amplitude of the signal increased so did the noise floor. With a 39Vpp input the noise floor rose to ~315nV/rtHz. In terms of the metric used to evaluate other ADC boards[5] this board comes in at 162dB.

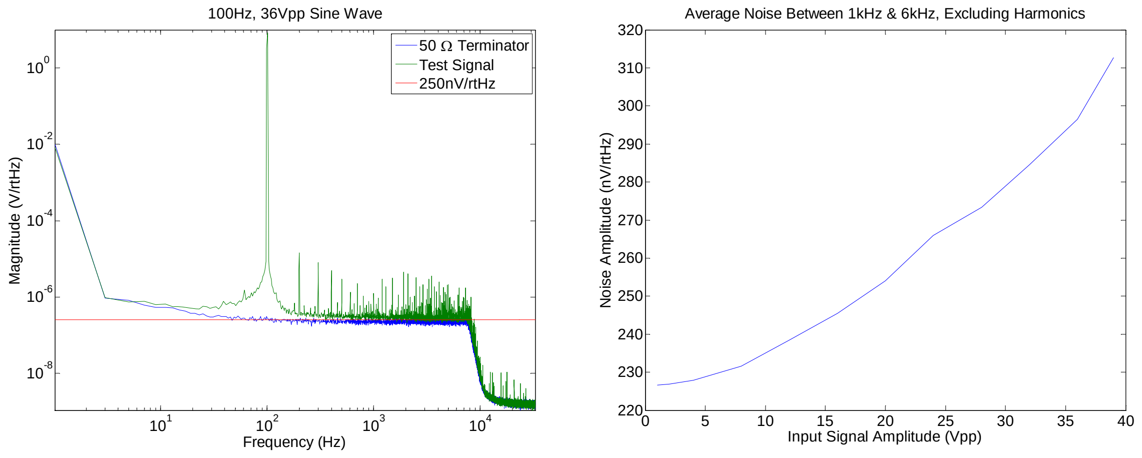


Figure 10

[5] T060158, Preliminary Noise Measurements of the General Standards PCI66-16AI64SSA-64-50M ADCs

4.3 Total Harmonic Distortion (THD)

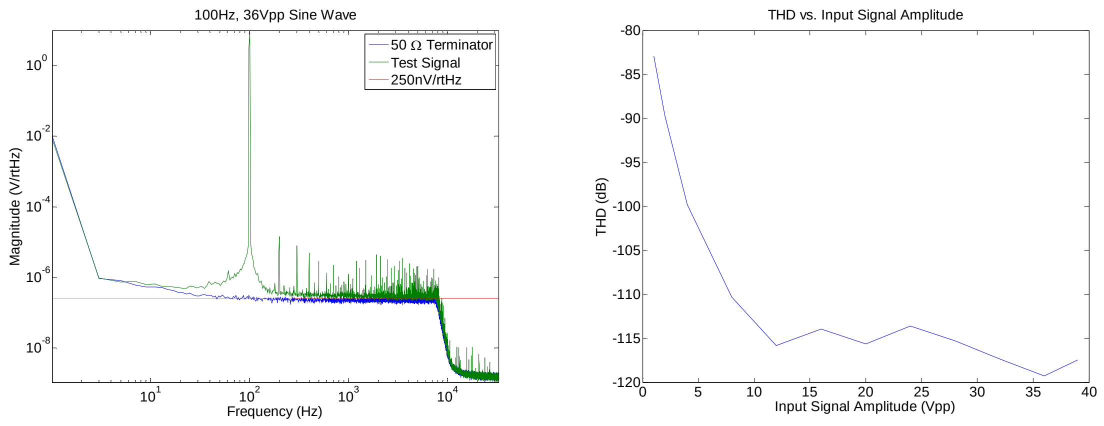


Figure 11

Injecting the 100Hz signal also provided a good opportunity to measure the total harmonic distortion. The power in a 2 Hz band centered on the first 5 harmonics was summed and compared to the power in the fundamental signal and the results are presented above. The THD specified for the AD7634 is -112dB[6] and the THD specified for the AP 2700 is -110dB[7] so the measurement is partially limited by the signal source.

4.4 Intermodulation Distortion (IMD)

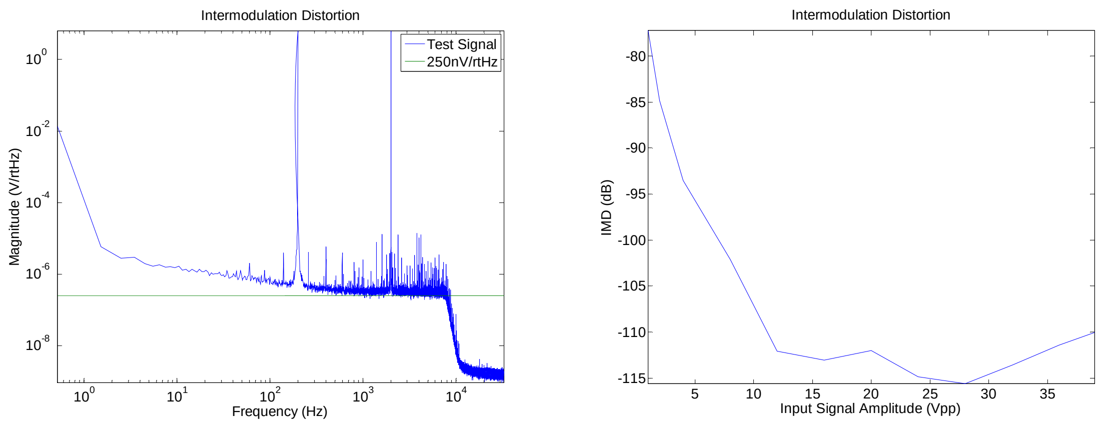
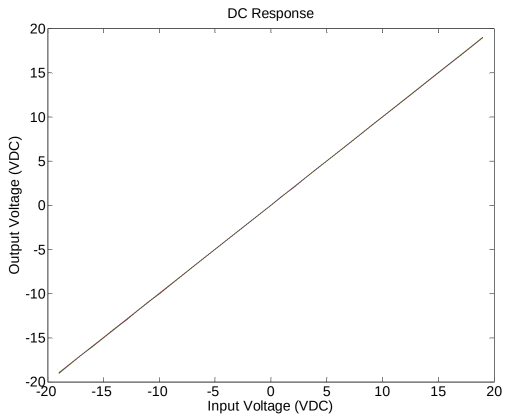


Figure 12

A similar test to measuring the THD is measuring the intermodulation distortion (IMD). This test is also called a duo tone test. The AP 2700 was set to IMD (D/A), SMPTE/DIN 1:1 with a high frequency of 2kHz and a low frequency of 200Hz. The AP 2700 has an IMD specification of -100dB so the measurement is again limited by the signal source. The THD of the 200Hz and 2kHz signal was similar to the THD of the 100Hz signal in the previous measurement.

[6] AD7634 Datasheet
[7] Audio Precision, Getting Started With Your 2700 Series Instrument

4.5 DC Response



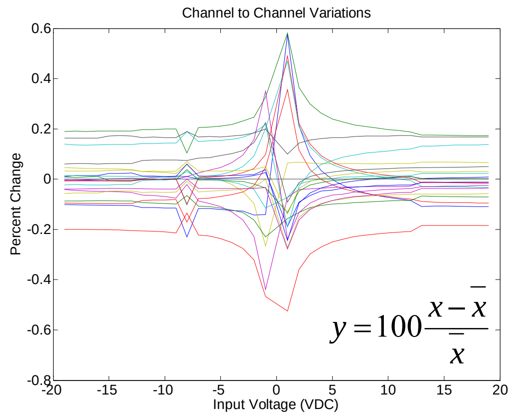
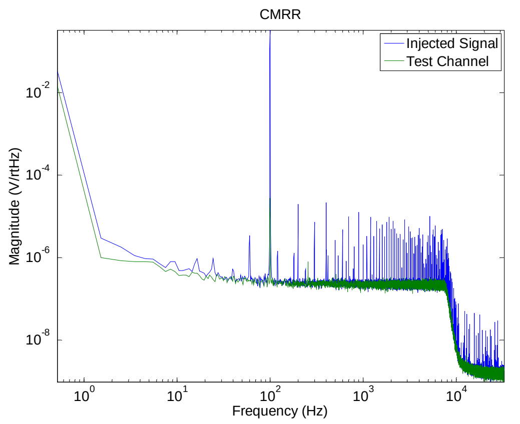


Figure 13

A DVC-350A voltage calibrator from Calibrators Inc. and a 9V battery were used to inject a DC voltage into all the channels simultaneously. The calibrator was used alone for voltages between -12 and +12 and the battery was added in series for voltages exceeding ±12 volts. Two seconds of data was collected for each channel and the mean value over that time was used to generate the plots above. For amplitudes greater than 5VDC the signal matching is 99.6% and for all amplitudes the signal matching is 99%. The measurement at -8VDC was taken at a later time than the other measurements and caused the jump seen above.

4.6 Common Mode Rejection Ratio



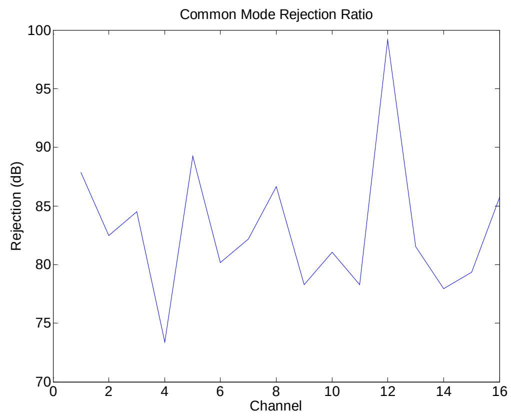


Figure 14

To measure the common mode rejection ratio (CMRR) a 1Vpp, 100Hz single ended signal was injected into both legs of the differential inputs of the channel being tested. The ground leg of the injected signal was connected to a ground testpoint on the board. The sum of the power in a 10Hz band centered on 100Hz was used to calculate the CMRR. The average across all 16 channels is 83dB.

4.7 Crosstalk

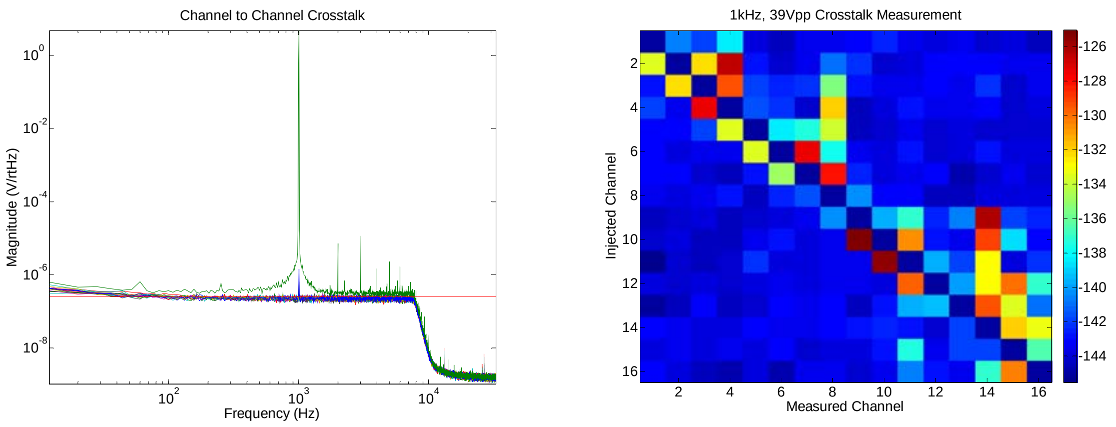


Figure 15

For this test the analog generator on the AP2700 was set to 1 kHz, 39 Vpp, normal, high accuracy sine wave with balanced, floating output. The signal was injected into one channel at a time and all the other channels were terminated with 50Ω. Eight seconds of data was collected for each channel then the signal was moved to the next channel. Eight more seconds were collected for each channel and the process continued until injections had been made on all 16 channels. The sum of the power in a 10Hz band centered on 1kHz was used to calculate the crosstalk. The maximum value across all the channels was -125dB and the mean value in the channels adjacent to the injection channels was -134dB. In the right hand plot of Figure 15 the channels along the diagonal were set to -145dB instead of 0dB to improve the contrast.

5 Conclusion

The prototype ADC board meets or exceeds all the design goals originally set forth. It presents an attractive alternative to commercial solutions currently available and also relaxes the requirements on other components in the input chain (photodiodes, whitening boards, etc.). The results of the tests are presented in the table below.

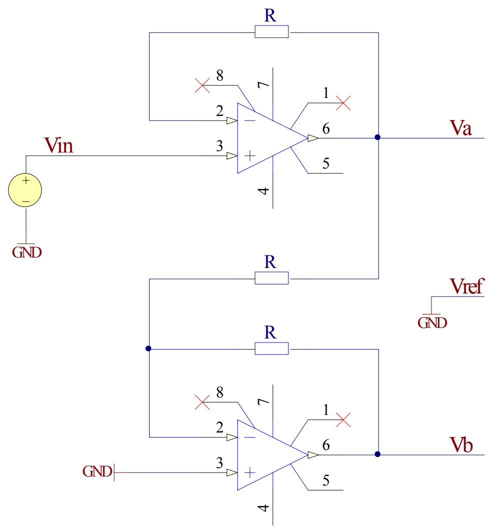
	Design Goal	Actual Value
THD	< -90dB	-110dB
Crosstalk	< -120dB	-125dB
CMRR	> 80dB	83dB

One final point worth mentioning is that the board was powered from Sorenson switching power supplies for all of these tests. There was also a small switching power supply[8] on the interface board. None of these supplies had a noticeable effect on the performance of the board. Also, this board is no more complicated than the boards currently used in LIGO and the manufacturing effort is no greater than these other boards.

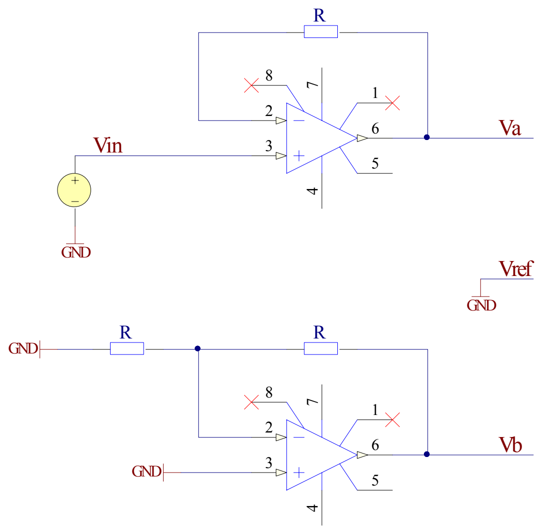
[8] PTH08T240W Datasheet

6 Appendix 1: Single Ended vs. Differential Peak to Peak Voltages

To alleviate any confusion this appendix defines how peak to peak voltages are measured for differential and single ended signals in this document. The output range is typically listed in the format ±X (Y Vpp). The first number, X, refers to the maximum and minimum output on a single leg with respect to a reference point (typically ground). The second number, Y, is the maximum signal on one leg minus the minimum signal on the second leg. The circuit snippet and example below further illustrates this.



Differential Driver



Single Ended Driver

<u>Differential Signal Example</u>

Va = Vin
Vb = -Va

Vp = max(Va) - min(Vb)

Vpp = 2*Vp

Vin = ±10
Vp = 20
Vpp = 40

<u>Single Ended Signal Example</u>

Va = Vin
Vb = 0

Vp = max(Va) - min(Vb)

Vpp = 2*Vp

Vin = ±10
Vp = 10
Vpp = 20

Appendix 2: Matlab Code

6.1 Data Acquisition

```
clear all
%run data to/from FPGA
%input arguments are: time(s), frequency(Hz), amplitude(fraction of full scale), ADC
input(1-4)
readme = [];
readme = strvcat(readme, ['50 ohm terminator, added 470u cap to +5.']);
readme = strvcat(readme, ['Ch. 0 - Original circuit.']);
readme = strvcat(readme, ['Ch. 1 - OPA134s installed in U1 & U2.']);
readme = strvcat(readme, ['Ch. 2 - OPA134s installed in U1, U2, & U3.']);
readme = strvcat(readme, ['Ch. 2 - OPA134s installed in U1, U2, U3, U4, & U5.']);
tic
istat = [];
ostat = [];
duration = 32;
chans = [4 17];

for k = 1:size(chans,2)

    disp(['Beginning call to FPGA for channel ' num2str(chans(k)) '.'])
    system(['InOut ' num2str(duration+2) ' 3000 .99 ' num2str(chans(k)) ' *
adctb_fpga_cclk.bit'])
    disp('Finished call to FPGA, loading data.')
    load output.txt;
%     load istatus.txt;
    load ostatus.txt;

    disp('Scaling and trimming output.')
    outputs(:,k) = output(65537:(duration+1)*65536)*20/2^26; %Trim the first and last
second off
%    istat = [istat; istatus];
    ostat = [ostat; ostatus];
    pause(1);
end

toc

figure(1), clf
for k = 1:size(chans,2)
    subplot(size(chans,2),1,k), plot(outputs(:,k))
    maxout = max(outputs(:,k));
    minout = min(outputs(:,k));
    ppout = maxout-minout;
    title(['Channel ' num2str(chans(k)) '   ' num2str([maxout minout ppout])])
end

figure(2), plot(ostat)

figure(5), plot(outputs)

AD1spec
```

6.2 Data Analysis

```
%spectrum of AD
clear f f_filt Amp Amp_filt Pyy Pyyfilt
sf = 2^16; %sampling frequency
dt = 1/sf;
N = size(outputs,1);
logbin = 0;
binsize = 32;

window = hann(N);

Tmax = (N*dt);
f = sf * (1:N) / (N);
f_filt = sDataCopy(f(1:N/2), binsize, logbin);

for k = 1:size(outputs,2)
    ftval = window.*outputs(:,k);
    Y = fft(ftval, N);
    Pyy = abs(Y).^2/(N*1.5)*8/sf;
    Pyyfilt(:,k) = sDataCopy(Pyy(1:N/2), binsize, logbin);
    Amp(:,k) = sqrt(Pyy);
    Amp_filt(:,k) = sqrt(Pyyfilt(:,k));
end

figure(3), clf
loglog(f_filt, Amp_filt, f_filt, 250e-9*ones(size(f_filt)))
legendtext = [];
for k = 1:size(chans,2)
    legendtext = strvcat(legendtext, ['Channel ' num2str(chans(k)+1)]);
end
legend(legendtext)

titlestr = ['Latest run']
title(titlestr);
xlabel('Frequency (Hz)');
ylabel('Magnitude (V/rtHz)');
%legend('ADC Test Board', 'SR785', '250nV/rtHz', 'Location', 'Northwest')
axis tight
clear Y ftval window
```

6.3 Noise & 100 Hz THD

```
clear all

width = 1;
files = cellstr(['100Hz_1Vpp_32s.mat ';...
                '100Hz_2Vpp_32s.mat ';...
                '100Hz_4Vpp_32s.mat ';...
                '100Hz_8Vpp_32s.mat ';...
                '100Hz_12Vpp_32s.mat';...
                '100Hz_16Vpp_32s.mat';...
                '100Hz_20Vpp_32s.mat';...
                '100Hz_24Vpp_32s.mat';...
                '100Hz_28Vpp_32s.mat';...
                '100Hz_32Vpp_32s.mat';...
                '100Hz_36Vpp_32s.mat';...
                '100Hz_39Vpp_32s.mat']);
for m = 1:12
    disp(['Loading ' files{m}])
    load(files{m})
    freqband = [];
    for n = 1000:100:6000
        freqband = [freqband find((f > n+width) & (f < n+100-width))];
    end
    %Find the average noise between 1k and 6k, ignoring harmonics.
    ave_noise(m) = sqrt(mean(Pyy(freqband)));

    freqband = find((f > 100-width) & (f < 100+width));
    %Find the power in the signal fundamental.
    THDden = sum(Pyy(freqband));

    %Find the power in the first five harmonics.
    for n = 200:100:600
        THDnum(n) = sum(Pyy(find((f > n-width) & (f < n+width))));
    end
     freqband = [find((f > 199) & (f < 201))...
         find((f > 299) & (f < 301))...
         find((f > 399) & (f < 401))...
         find((f > 499) & (f < 501))...
         find((f > 599) & (f < 601))];
     THDnum = sum(Pyy(freqband));

    THD(m) = 10*log10(THDnum/THDden);
end

figure(1), clf
plot([1 2 4 8 12 16 20 24 28 32 36 39], ave_noise*1e9)
title('Average Noise Between 1kHz & 6kHz, Excluding Harmonics')
xlabel('Input Signal Amplitude (V)')
ylabel('Noise Amplitude (nV/rtHz)')

figure(2), clf
plot([1 2 4 8 12 16 20 24 28 32 36 39], THD)
title('THD vs. Input Signal Amplitude')
xlabel('Input Signal Amplitude (V)')
ylabel('THD (dB)')
```

6.4 DC Response

```
clear all

files = cellstr(['n19VDC_2s.mat';...
                 'n18VDC_2s.mat';...
                 'n17VDC_2s.mat';...
                 'n16VDC_2s.mat';...
                 'n15VDC_2s.mat';...
                 'n14VDC_2s.mat';...
                 'n13VDC_2s.mat';...
                 'n12VDC_2s.mat';...
                 'n11VDC_2s.mat';...
                 'n10VDC_2s.mat';...
                 'n9VDC_2s.mat ';...
                 'n8VDC_2s.mat ';...
                 'n7VDC_2s.mat ';...
                 'n6VDC_2s.mat ';...
                 'n5VDC_2s.mat ';...
                 'n4VDC_2s.mat ';...
                 'n3VDC_2s.mat ';...
                 'n2VDC_2s.mat ';...
                 'n1VDC_2s.mat ';...
                 '0VDC_2s.mat  ';...
                 'p1VDC_2s.mat ';...
                 'p2VDC_2s.mat ';...
                 'p3VDC_2s.mat ';...
                 'p4VDC_2s.mat ';...
                 'p5VDC_2s.mat ';...
                 'p6VDC_2s.mat ';...
                 'p7VDC_2s.mat ';...
                 'p8VDC_2s.mat ';...
                 'p9VDC_2s.mat ';...
                 'p10VDC_2s.mat';...
                 'p11VDC_2s.mat';...
                 'p12VDC_2s.mat';...
                 'p13VDC_2s.mat';...
                 'p14VDC_2s.mat';...
                 'p15VDC_2s.mat';...
                 'p16VDC_2s.mat';...
                 'p17VDC_2s.mat';...
                 'p18VDC_2s.mat';...
                 'p19VDC_2s.mat']);

for m = 1:39
    disp(['Loading ' files{m}])
    load(files{m})
    meanout(m,:) = mean(outputs);
end

measout = [-19.006 -18.007 -17.006 -16.007 -15.007 -14.008 -13.007 -12.000 ...
    -11.0001 -9.9996 -8.9998 -8.0004 -7.0005 -6.0004 -5.0003 -4.0002 -3.0001 ...
    -1.9996 -0.99975 0.000293 0.99982 2.00000 3.0003 4.0004 5.0005 6.0005 ...
    7.0004 8.0002 8.9997 9.9994 11.000 11.999 13.007 14.007 15.007 16.006 ...
    17.006 18.006 19.006];
figure(1), plot([-19:19], meanout)
title('DC Response')
xlabel('Input Voltage (VDC)'), ylabel('Output Voltage (VDC)')

for k = 1:21
    ratioout(:,k) = (meanout(:,k)-meanout(:,21))./meanout(:,21);
end
figure(2), clf, plot([-19:-1 1:19],100*ratioout([1:19 21:39],:))
title('Channel to Channel Variations')
xlabel('Input Voltage (VDC)'), ylabel('Percent Change')
text(8, -0.5, 'y = $\frac{x-\bar x}{\bar x}$','interpreter','latex', 'fontsize', 20)
```

6.5 Common Mode Rejection Ratio

```
clear all

 files = cellstr(['100Hz_1Vpp_32s_ch0-3CMRR.mat  ';...
                  '100Hz_1Vpp_32s_ch4-7CMRR.mat  ';...
                  '100Hz_1Vpp_32s_ch8-11CMRR.mat ';...
                  '100Hz_1Vpp_32s_ch12-15CMRR.mat']);
CMRR = [];

 for m = 1:4
    disp(['Loading ' files{m}])
    load(files{m})
    freqband = find((f_filt > 95) & (f_filt < 105));
%    CMRR = [CMRR 20*log10(max(Amp_filt(freqband,2:5))/max(max(Amp_filt(freqband,1))))];
    CMRR = [CMRR 10*log10(sum(Pyyfilt(freqband, 2:5))/sum(Pyyfilt(freqband,1)))];
 end

figure(1), clf, loglog(f_filt, Amp_filt(:,1:2))
axis tight, xlabel('Frequency (Hz)'), ylabel('Magnitude (V/rtHz)')
legend('Injected Signal', 'Test Channel')
title('CMRR')

figure(2), clf, plot(-CMRR)
title('Common Mode Rejection Ratio')
xlabel('Channel')
ylabel('Rejection (dB)')
```

6.6 Crosstalk

```
clear all

 files = cellstr(['1kHz_39Vpp_8s_ch0.mat ';...
     '1kHz_39Vpp_8s_ch1.mat ';...
     '1kHz_39Vpp_8s_ch2.mat ';...
     '1kHz_39Vpp_8s_ch3.mat ';...
     '1kHz_39Vpp_8s_ch4.mat ';...
     '1kHz_39Vpp_8s_ch5.mat ';...
     '1kHz_39Vpp_8s_ch6.mat ';...
     '1kHz_39Vpp_8s_ch7.mat ';...
     '1kHz_39Vpp_8s_ch8.mat ';...
     '1kHz_39Vpp_8s_ch9.mat ';...
     '1kHz_39Vpp_8s_ch10.mat';...
     '1kHz_39Vpp_8s_ch11.mat';...
     '1kHz_39Vpp_8s_ch12.mat';...
     '1kHz_39Vpp_8s_ch13.mat';...
     '1kHz_39Vpp_8s_ch14.mat';...
     '1kHz_39Vpp_8s_ch15.mat']);


 for m = 1:16
    disp(['Loading ' files{m}])
    load(files{m})
    freqband = find((f_filt > 995) & (f_filt < 1005));
    for n = 1:16
        xtalk(m,n) = 10*log10(sum(Pyyfilt(freqband, n))/sum(Pyyfilt(freqband,m)));
    end
 end

for m = 1:16
    xtalk(m,m) = -145;
end

figure(1), imagesc(xtalk), colorbar
title('1kHz, 39Vpp Crosstalk Measurement')
xlabel('Measured Channel')
ylabel('Injected Channel')

%Find max crosstalk
max(xtalk(find(xtalk < 0)))
xtalkmean = xtalk(1,2) + xtalk(16,15);
for m = 2:15
    xtalkmean = xtalkmean + xtalk(m,m-1) + xtalk(m,m+1);
end
xtalkmean = xtalkmean/30

figure(2), loglog(f_filt(2:end), Amp_filt(2:end,:), f_filt(2:end), 250e-
9*ones(size(f_filt(2:end))))
axis tight, xlabel('Frequency (Hz)'), ylabel('Magnitude (V/rtHz)')
title('Channel to Channel Crosstalk')
```

6.7 Intermodulation Distortion

```
clear all
files = cellstr(['200Hz_2kHz_1Vpp_32s_ch15IMD.mat   ',
                 '200Hz_2kHz_2Vpp_32s_ch15IMD.mat   ',
                 '200Hz_2kHz_4Vpp_32s_ch15IMD.mat   ',
                 '200Hz_2kHz_8Vpp_32s_ch15IMD.mat   ',
                 '200Hz_2kHz_12Vpp_32s_ch15IMD.mat  ',
                 '200Hz_2kHz_16Vpp_32s_ch15IMD.mat  ',
                 '200Hz_2kHz_20Vpp_32s_ch15IMD.mat  ',
                 '200Hz_2kHz_24Vpp_32s_ch15IMD.mat  ',
                 '200Hz_2kHz_28Vpp_32s_ch15IMD.mat  ',
                 '200Hz_2kHz_32Vpp_32s_ch15IMD.mat  ',
                 '200Hz_2kHz_36Vpp_32s_ch15IMD.mat  ',
                 '200Hz_2kHz_39Vpp_32s_ch15IMD.mat  ']);

CMRR = [];

f1 = 2000;
f2 = 200;
width = 0.2;

IMDbands = [f1-f2 f1+f2 f1-2*f2 2*f1-f2 f1+2*f2 2*f1+f2...
    f1-3*f2 2*f1-2*f2 3*f1-f2 f1+3*f2 2*f1+2*f2 3*f1+f2];
THDf1bands = [2*f1 3*f1 4*f1 5*f1 6*f1];
THDf2bands = [2*f2 3*f2 4*f2 5*f2 6*f2];

for m = 1:size(files,1)
    disp(['Loading ' files{m}])
    load(files{m})

    %Calculate the power in the carrier frequencies.
    freqband = find((f > f1-width) & (f < f1+width));
    f1pow = sum(Pyy(freqband));

    freqband = find((f > f2-width) & (f < f2+width));
    f2pow = sum(Pyy(freqband));

    %Calculate the power in the intermodulation difference frequencies.
    IMDpow = 0;
    for n = 1:size(IMDbands,2)
        freqband = find((f > IMDbands(n)-width) & (f < IMDbands(n)+width));
        IMDpow = IMDpow + sum(Pyy(freqband));
    end

    %Calculate the THD for f1
    THDf1pow = 0;
    for n = 1:size(THDf1bands,2)
        freqband = find((f > THDf1bands(n)-width) & (f < THDf1bands(n)+width));
        THDf1pow = THDf1pow + sum(Pyy(freqband));
    end

    %Calculate the THD for f2
    THDf2pow = 0;
    for n = 1:size(THDf2bands,2)
        freqband = find((f > THDf2bands(n)-width) & (f < THDf2bands(n)+width));
        THDf2pow = THDf2pow + sum(Pyy(freqband));
    end

    THDf1(m) = 10*log10(f1pow/THDf1pow);
    THDf2(m) = 10*log10(f2pow/THDf2pow);
    IMD(m) = 10*log10((f1pow+f2pow)/IMDpow);
end

v = [1 2 4 8 12 16 20 24 28 32 36 39];
figure(1), plot(v, -IMD)
axis tight, xlabel('Input Signal Amplitude (Vpp)'), ylabel('IMD (dB)')
title('Intermodulation Distortion')

figure(2), loglog(f_filt, Amp_filt, f_filt, 250e-9*ones(size(f_filt)))
 axis tight, xlabel('Frequency (Hz)'), ylabel('Magnitude (V/rtHz)')
title('Intermodulation Distortion')
```